Phase-mode pipelined parallel multiplier

doi: 10.1109/77.919402

Phase-Mode Pipelined Parallel Multiplier

Takeshi Onomi, Kiyoshi Yanagisawa, Masashi Seki, and Koji Nakajima

***Abstract*—We propose a pipelined parallel multiplier in phase-mode logic. The multiplier can be composed of combinations of gates which are the basic devices of the phase-mode logic. Experimental operations of the ICF gate and the Adder cell for the multiplier are reported. The proposed multiplier has a Wallace-tree structure comprising trees of carry save adders for the addition of partial products. This structure has a regular layout, hence it is suitable for a pipeline scheme. In the final stage of multiplication, a fast carry lookahead adder is used for generating a multiplication result. Using a Verilog-HDL simulation, we show that the parallel multiplier with 2.5kA/cm^2 Nb/AlOx/Nb junctions can operate over 10 GHz.**

***Index Terms*—multiplier, phase mode, pipeline, single flux quantum,**

I. Introduction

RECENTLY, requirements of high-speed computation are increasing on the various areas such as nuclear science, weather forecasting, information processing of communication systems, molecular science, development of complex computer systems, etc. A single-flux-quantum (SFQ) logic has a great potential for such high-speed digital computations [1]-[3]. We have proposed phase-mode logic, which is an SFQ logic for a digital computation.

In this paper, we propose a pipelined parallel multiplier in phase-mode logic. Firstly, we report experimental results of the basic gate of phase-mode logic and the adder cell which are used for designing the multiplier. Secondly, a design of a carry lookahead (CLA) adder is described. This CLA adder plays an important role in the final stage of multiplication. Thirdly, we propose a multiplier having trees of carry save adders for the addition of partial products. This structure has a regular layout, hence it is suitable for a pipeline scheme. Finally, we discuss the performance of the multiplier. Using a Verilog-HDL simulation, we show that the parallel multiplier with 2.5kA/cm^2 Nb/AlOx/Nb junctions can operate over 10 GHz.

II. ICF Gate and Adder Cell

In this section, we describe the experimental test results of the basic gate and the adder cell which are the most basic circuits of pipelined parallel adder and multiplier.

A. *ICF (INHIBIT controlled by fluxon) gate*

We have proposed ICF(INHIBIT controlled by fluxon) gate as the basic device of the phase-mode logic. Since the first proposal, some types of ICF gates have been proposed. In this section, we describe the ICF gate designed by RSFQ scheme.

Fig. 1 shows the ICF gate and its Moore diagram. This gate has the same structure of D Flip-Flop with complementary outputs in RSFQ logic [4]. However, some modifications of the D Flip-Flop are needed to realize the perfect operation of the ICF gate. The modifications are the addition of Reset and C terminals. The operations of ICF gate are as follows: •If the internal state of the gate is "0", that is, no SFQ is in the loop including J2, J3, J7, L2, and J6, an SFQ from X outputs to A terminal through J1, J2, J7, and J5. •If the internal state is "0", an SFQ from Y is stored in the loop consisting of J2, J3, J7, L2, and J6 after passing through J11 and J7. •If the internal state is "1", an SFQ from X is combined with the stored SFQ, and outputs to B by J1, J3, J4, and J8 switching. •If the internal state is "1", an SFQ from Re is combined with the stored SFQ, and outputs to C by J10 and J6 switching. •If the internal state is "0", an SFQ from Re is emitted by J9.

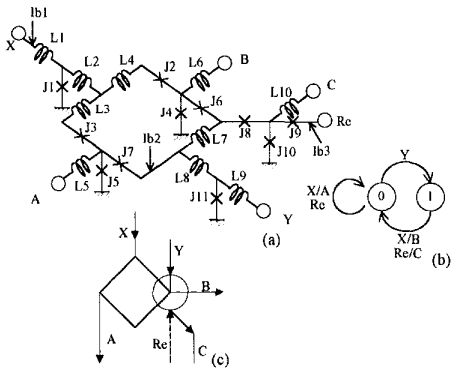

Fig.1. ICF gate. (a) Circuit diagram. The device parameters are Ic1 = 0.35mA, Ic2 = 0.28mA, Ic3 = 0.34mA, Ic4 = Ic5 = 0.25mA, Ic6 = 0.31mA, Ic7 = Ic8 = 0.20mA, Ic9 = 0.27mA, Ic10 = 0.12mA, L1 = 0.8pH, L2 = 0.7pH, L3 = L4 = 0.1pH, L5 = 4.0pH, L6 = 1.5pH, L7 = 3.1pH, L8 = 2.9pH, L9 = 1.2pH, L10 = 3.0pH, Ib1 = 0.23mA, Ib2 = 0.19mA, and Ib3 = 0.21mA. (b) Moore diagram. (c) Symbol.

Figs. 2(a) and (b) show the microphotograph and the functional test result of the ICF gate, respectively. The gate is fabricated by using NEC standard 2.5kA/cm^2 Nb/AlOx/Nb process. This result shows the proper operation of ICF gate. However, the measured bias margin is very narrow due to the dispersion of circuit parameters

B. *Adder Cell*

A serial input adder cell is achieved by feeding A output of an ICF gate back to Y input. Fig. 3 shows the adder cell using an ICF gate. Fig. 4 shows the low-speed test result of a

Manuscript received Sept. 18, 2000. This work was supported in part by Special Coordination Funds for promoting Science and Technology of the Science and Technology Agency of the Japanese Government.

T. Onomi is with L.E.I.S., Research Institute of Electrical Communication, Tohoku University, Sendai 980-8577 Japan (telephone: +81-22-217-5559, e-mail: onomi@riec.tohoku.ac.jp).

K. Yanagisawa was with L.E.I.S., Research Institute of Electrical Communication, Tohoku University, Sendai 980-8577 Japan. He is now with Nippon Signal Co., (telephone: +81-48-859-2921, e-mail: yanagi-k@signal.co.jp).

K. Nakajima is with L.E.I.S., Research Institute of Electrical Communication, Tohoku University, Sendai 980-8577 Japan (telephone: +81-22-217-5558, e-mail: hello@riec.tohoku.ac.jp).

fabricated adder cell. This result shows the proper operation of the adder cell. The measured bias margin is ±7% which is lower than designed value(±39%).

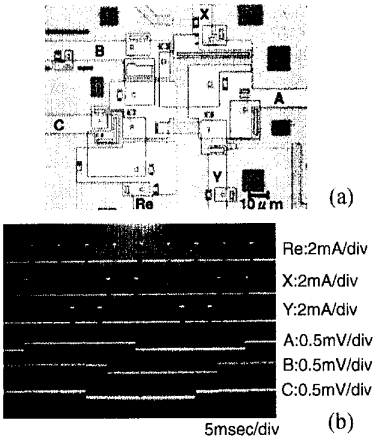

Fig. 2. ICF gate fabricated by NEC 2.5kA/cm^2 Nb/AlOx/Nb standard process. (a) Microphotograph. (b) Low-speed test result. Three lower traces show output voltages of the SFQ/DC(T-flip-flop) pulse detectors.

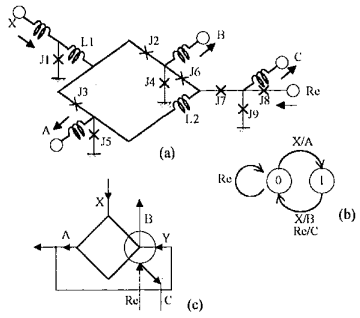

Fig. 3. Adder cell. (a) Circuit diagram. (b) Moore diagram. (c) Symbol.

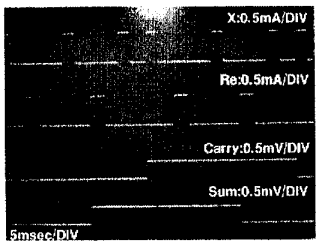

Fig. 4. Low-speed test result of the fabricated adder circuit.

III. Carry Lookahead Adder

A carry lookahead (CLA) adder described in this section plays an important role in a final stage of multiplication. In phase-mode logic, a parallel ripple carry adder has been proposed [1]. A ripple carry adder has the simplest structure, however it has a weak point of an increase of the operating time which is proportional to the number of bits. Most of parallel-adder circuits using semiconductor devices are based on a carry-lookahead architecture. In the RSFQ logic, a fast pipelined parallel CLA using only two types of cells (inverter and D-flip-flop) has been proposed [5]. A CLA adder has regularity in its structure and therefore it is suitable for a pipelined scheme. In this section, we describe a design of the carry lookahead adder using the ICF gates and adder cells described in the previous section. The CLA adder includes three arithmetical blocks which are Preprocessing, Carry Lookahead, and Postprocessing. In the following sections, the timing of the system is controlled by a traditional method of Phase-Mode logic [1]. The system operates asynchronously by using one timing signal being attached to one word. This method has the disadvantage of relatively lower-speed operation than synchronized circuits. However, this method is simple and does not need much attention to timing design. Timing signals to each of the bits on a pipeline stage are provided by signal distribution trees.

A. *Preprocessing*

$A(a_n a_{n-1}\cdots a_2 a_1)$, $B(b_n b_{n-1}\cdots b_2 b_1)$, and $C(c_n c_{n-1}\cdots c_2 c_1)$ denote N bits augend, addend, and carry, respectively. The Preprocessing block generates the P(propagate) signal and the G(generate) signal represented by following equations

$$g_i = a_i \cdot b_i \tag{1}$$

$$p_i = a_i \oplus b_i \quad (\text{or} \quad p_i = a_i + b_i). \tag{2}$$

Namely, g_i=1 represents the generation of the carry c_i , and p_i=1 represents the generation of the carry c_i when c_{i-1}=1. P and G signals can be generated by a PG generator shown in Fig. 5. Q signal (q_i=a_i⊕b_i) in Fig.5 is used for generating Sum in the Postprocessing block.

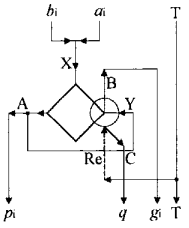

Fig. 5. PG generator.

B. *Carry Lookahead*

This block generates carry signals by using P and G signals. The carry c_i is represented by equation

$$c_i = g_i + p_i \cdot c_{i-1}. \tag{3}$$

In a block including continuous bits from bit j to bit i, P and G signals can be defined as $p_{i:j}$ and $g_{i:j}$. $p_{i:j}$ signifies that a carry will propagate from bit j to bit i. Similarly $g_{i:j}$ denotes that a carry is generated in at least one of the bit positions from j to i inclusive and propagated to bit position i. The carry c_i=$g_{i:0}$ can be calculated efficiently by using the operator (Δ) introduced by Brent and Kung [6]. The operator is used:

$$(p_{i:j}, g_{i:j}) = (p_{i:k+1}, g_{i:k+1}) \Delta (p_{k+1:j}, g_{k+1:j}) \tag{4}$$

which is defined as

$$p_{i:j} = p_{i:k+1} p_{k+1:j} \quad \text{and} \quad g_{i:j} = g_{i:k+1} + p_{i:k+1} g_{k:j}. \tag{5}$$

The calculations of P=P_a· P_b and G=G_a+G_b·P_a can be achieved by using three ICF gates as shown in Fig.6.

Various carry calculation methods using the Δ operator have been proposed. Fig.7 shows the some methods having a tradeoff between speed and number of circuit elements. Fig.7(a) shows the notations of the Δ operator cell and the dummy cell having only the data shift function. Fig.7(b) and (c) are Sklansky's method [7] and Kogge and Stone's method [8], respectively. These methods have small stages of pipeline, hence they can achieve high-speed carry calculations. On the other hand, Brent and Kung's method [6] shown in Fig.7(d)

has small circuit elements. Table I shows the comparison of these methods in a 32-bit CLA.

C. *Postprocessing*

A sum of bit i (s_i) is obtained by equation

$$s_i = p_i \oplus c_{i-1}. \quad (6)$$

Using q_i generated by Preprocessing, a sum can be calculated as $s_i=q_i \oplus g_{i-1}$. This operation can be achieved by using Sum generator shown in Fig.8.

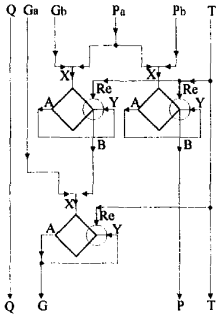

Fig. 6. Δ operator cell.

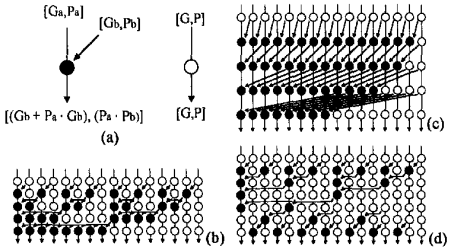

Fig. 7. Computation methods of CLA. (a) Notation. A filled dot and a circle denote a Δ operator cell and a shift operator cell, respectively. (b) Sklansky's method. (c) Kogge-Stone's method. (d) Brent-Kung's method.

TABLE I
COMPARISON OF 32-BIT CLA ALGORITHMS

Scheme	Depth of tree	●	○
Sklansky	5	80	80
Kogge-Stone	5	129	31
Brent-Kung	9	57	199

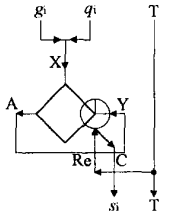

Fig. 8. Sum generator.

IV. PIPELINED PARALLEL MULTIPLIER

In this section, we describe a design of the multiplier with a Wallace-tree structure [9]. This structure has a simple and regular layout therefore can easily comprise the combinations of ICF gates and adder cells in the previous section. The multiplier includes three arithmetical blocks which are a generation of partial products, an addition of partial products, and a generation of the multiplication result.

A. *Generation of partial products*

An AND cell shown in Fig. 10 is used for generating a partial product of a multiplication. AND cells forms an AND array shown in Fig.11. The AND array sends SFQs (partial products) to each of bit lines serially.

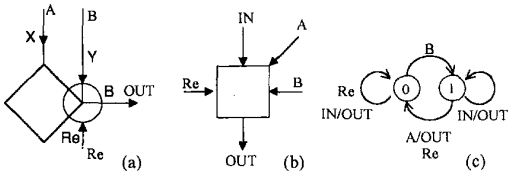

Fig. 10. AND cell. (a)Symbol of the cell using ICF gate. (b)Notation. (c)Moore diagram.

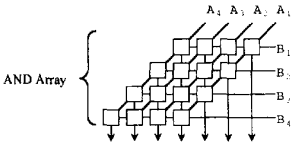

Fig. 11. AND array(4-bit).

B. *Addition of partial products*

We propose a multiplier with a tree structure using a carry save adder(CSA). The CSA can be realized by using a full adder circuit with an additional adder cell to store the carry output as shown in Fig.12. The CSA has usually three inputs and two outputs. The CSA proposed in this section can easily expand into the cell with more inputs as shown in Fig.12(b). After a partial products input to the CSA, the result of the addition is sent to outputs by reset signal. Fig.13 shows the CSA cell using an ICF gate. Fig.14 shows a 7-input CSA array which varies the addition of seven numbers into the one of three numbers.

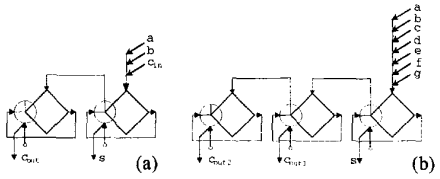

Fig. 12. (n,m)CSA. n and m denote input number and output number, respectively. (a) (3,2)CSA. (b) (7,3)CSA.

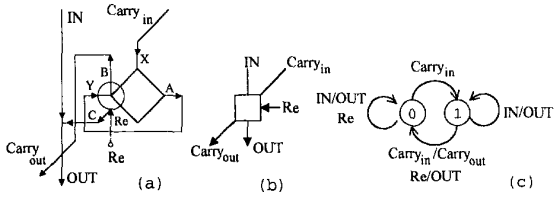

Fig. 13. CSA cell. (a)Symbol of the cell using ICF gate. (b)Notation. (c)Moore diagram.

C. *Generation of multiplication result*

After additions of partial products are executed in order, the addition of two numbers remains finally. Using the carry lookahead adder in the previous section, the multiplication result can be obtained.

Fig.15 shows an example of a 32×32-bit multiplier using (7,3) CSA.

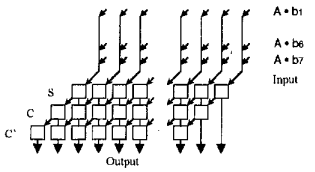

Fig. 14. 7-input CSA array

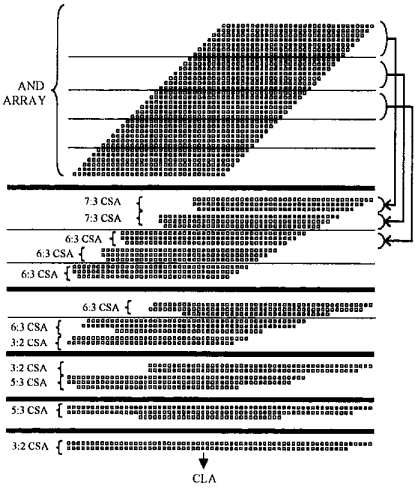

Fig. 15. Scheme of 32-bit multiplier using 7-input CSA array. Fine horizontal lines show the partitions of the input of next CSA array. Thick horizontal lines show the partitions of the operation of the multiplication. The final stage of the CSA array is connected to the CLA adder explained in section III.

V. Estimations of Multiplication Performance

The delay time of CSA is related to a time O(n) being proportional to a number of serial inputs and a time O(log(n)) depending on bit number of the adder. Accordingly, the processing time of one-stage CSA is represented by

$$T = nT_{in} + mT_{add}, \qquad (6)$$

where n is a number of inputs, m is a bit number of the adder, T_{in} is a time interval between input pulses, and T_{add} is a delay time of carry propagation per one bit. If the multiplier is designed by using a 3-input CSA array, the processing time per stage of the pipeline is minimum, therefore, the throughput is maximum. However, the tree structure is large and complicated because of an increase of the number of pipeline stages. On the other hand, if the number of inputs is increased, the scale of trees can be reduced by the decrease of pipeline stages. However, the throughput is decreased. Namely, it means a tradeoff between operation speed and integration scale. While we can not expect the maximum throughput, the integration scale of CSA trees can be decreased by using a 7-input CSA. Table II, Table III and Table IV shows the estimations of 32-bit multiplier without CLA block, integration scale, and estimations of Kogge-Stone CLA block, respectively. These estimations were carried out by Verilog-HDL simulations assuming J_c = 2.5kA/cm^2. A final multiplication result was obtained by a 64-bit Kogge-Stone CLA adder. The maximum throughput of the CLA is estimated to be over 15 GOPS by the Verilog-HDL simulation. Hence, total throughput of the multiplier is limited by the throughput of CSA trees shown in Table II. As a result, a 32-bit multiplication over 10 GHz can be achieved.

TABLE II
Estimations of 32-bit Multiplier without CLA Block

Maximum input number of CSA	Depth of the tree	Throughput (GOPS : gigaoperation per second)
3	9	13.3
7	5	6.7
32	3	2.3

TABLE III
Integration Scale

Block	Number of ICF gates
AND Array	1024
32-input CSA Tree	561
7-input CSA Tree	1110
3-input CSA Tree	2541

TABLE IV
Estimations of Kogge-Stone CLA Block

Word length	Depth of the tree	Delay(ps)
8bit	3	304
16bit	4	375
32bit	5	446
64bit	6	517

VI. Conclusion

We have proposed a pipelined parallel multiplier using phase-mode logic. The multiplier is comprised of combinations of ICF gates which are the basic devices of the phase-mode logic. Experimental operations of the ICF gate and the Adder cell for the multiplier have been confirmed. The proposed multiplier has a Wallace-tree structure comprising trees of carry save adders for the addition of partial products. This structure has regularity in its layout, hence it is suitable for a pipelined scheme. On the final stage of multiplication, a fast carry lookahead adder is used for generating a multiplication result. Using a Verilog-HDL simulation, we have shown that the parallel multiplier with 2.5 kA/cm^2 Nb/AlOx/Nb junctions can achieve fast multiplication over 10 GHz.

References

[1] K. Nakajima, H. Mizusawa, H. Sugahara, and Y. Sawada, "Phase-mode Josephson computer System," *IEEE Trans. Appl. Superconduct.*, vol.1, pp.29-36, March 1991.

[2] K. K. Likharev and V. K. Semenov, "RSFQ logic/memory family: A new Josephson-junction technology for sub-terahertz-clock-frequency digital systems," *IEEE Trans. Appl. Superconduct.*, vol.1, pp.3-28, March 1991.

[3] M. Hosoya, W. Hioe, J. Casas, R. Kamikawai, Y. Harada, Y. Wada, H. Nakane, R. Suda, and E. Goto, "Quantum flux parametron: A single quantum flux device for Josephson supercomputer," *IEEE Trans. Appl. Superconduct.*, vol.1, pp.77-89, June 1991.

[4] A. F. Kirichenko, V. K. Semenov, Y. K. Kwong, and V. Nandakumar, "4-bit rapid single-flux-quantum decoder," *IEEE Trans. Appl. Superconduct.*, vol.5, pp.2857-2860, June 1995.

[5] P. Bunyk and P. Litskevitch, "Case study in RSFQ design: Fast pipelined Parallel Adder," *IEEE Trans. Appl. Superconduct.*, vol.9, pp.3714-3720, June 1999.

[6] R. Brent and H. Kung, "A regular layout for parallel adders," *IEEE Trans. Comput.*, vol.C-31, pp.260-264, March 1982.

[7] J. Sklansky, *IRE Trans. Electronic Comput.*, vol.9, p.226, 1960.

[8] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Trans. Comput.*, vol.C-22, pp.786-792, August 1973.

[9] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Electron. Comp.*, vol. EC-13, pp. 14-17, Feb. 1964.